OO Basic Concepts

BIRZEIT UNIVERSITY

By: Mamoun Nawahdah (Ph.D.)
2022

Problems with Procedural Languages

❖ Data does not have an owner.

❖ Difficult to maintain data integrity.

❖ Functions are building blocks.

❖ Many functions can modify a given block of data.

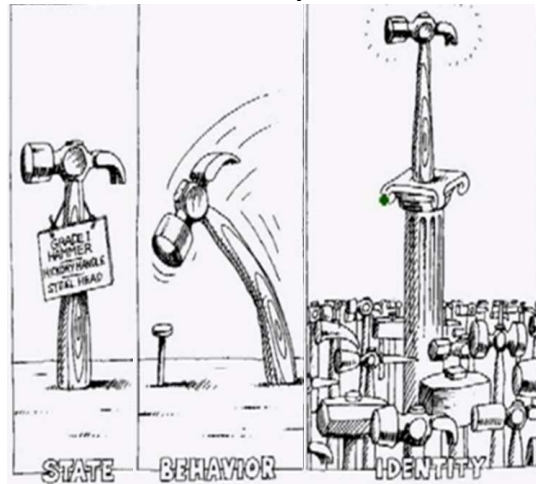❖ Difficult to trace bug sources when data is corrupted.

What is **O**bject?

❖ A thing that has **state**, exhibits some well defined **behavior**, and has a unique **identity**.

❖ **State**:
- Data members
- Fields
- Properties

❖ **Behavior**:
- Member functions
- Methods

Abstraction - Modeling

❖ **Abstraction** focuses upon the **essential** characteristics of some object, relative to the perspective of the viewer.

Abstraction focuses upon the essential characteristics of some object, relative to the perspective of the viewer.

An abstraction includes the essential details relative to the perspective of the viewer
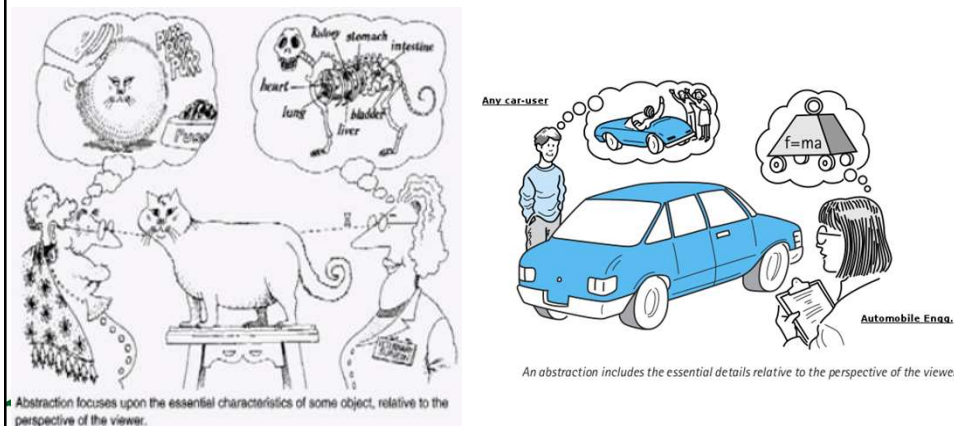
What is **Class**?

❖ A **class** represents a set of objects that share common structure and a common behavior.

❖ A **class** is a **blueprint** or **prototype** that defines the variables and methods common to all objects of a certain kind.
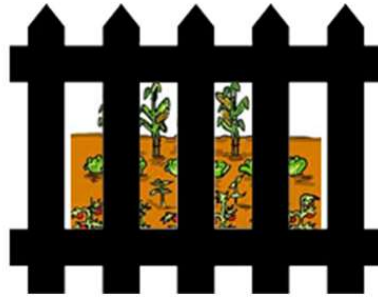
Class (Car)	
Attributes	Methods
Company	Move
Type	Stop
Model	Turn Right
Color	Turn Left
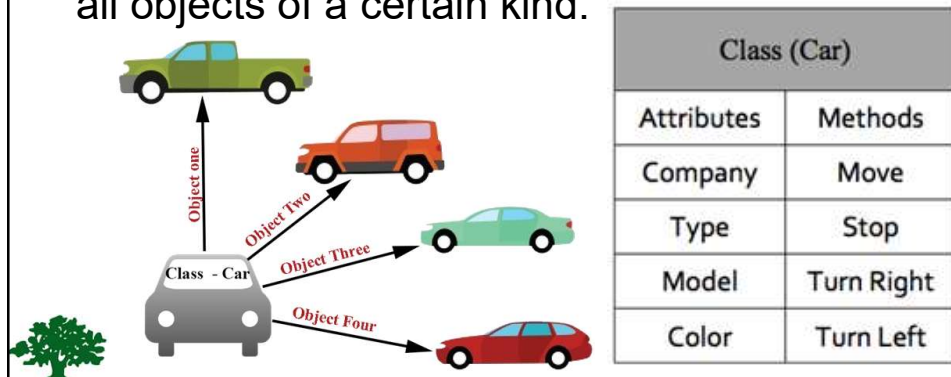
Class Access

❖ **Problem**: You have a garden and it is **public**. Anyone can take the properties of the garden when they want.

Class Access cont.

❖ **Solution**: Make it **private**, put a high fence around my garden, now it is safe!
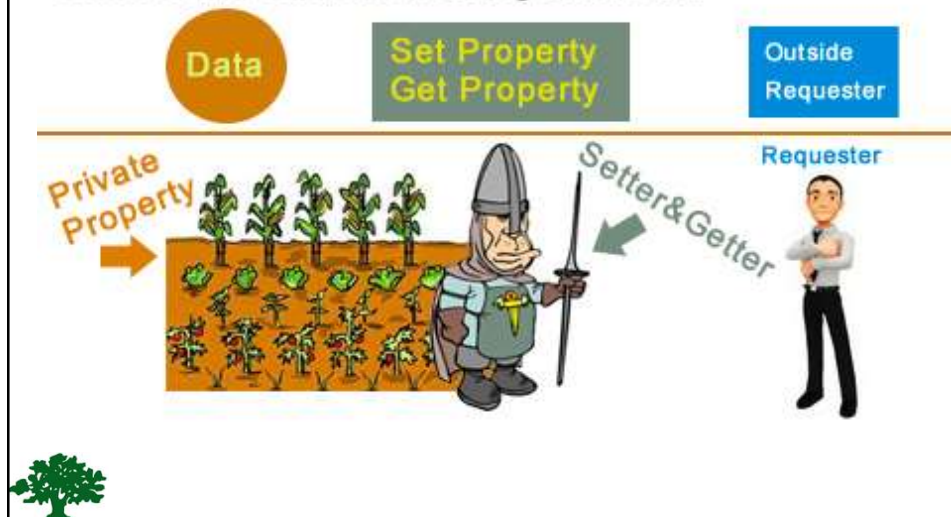

❖ But wait, I can no longer access my own garden!!!!

Class Access cont.

❖ **Solution**: Hire a private guard and give him **rules** on who is able to access the garden.
❖ Anyone want use the garden must get permission from guard.
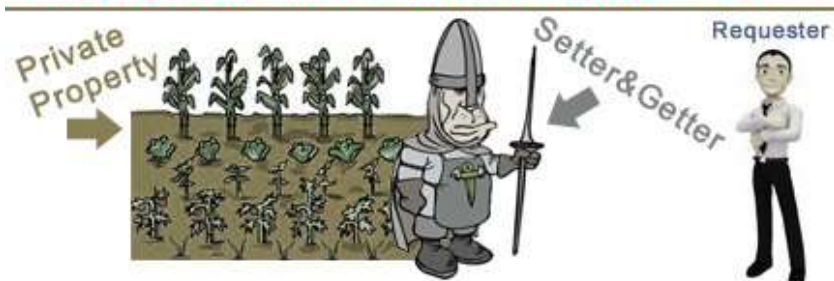❖ Garden is now **safe** and **accessible**.

4

Class Access cont.

Setters and Getters to Safeguard Data

Initialization of Objects

What if garden had weeds from the beginning?

❖ **Constructors** ensure correct initialization of all data. They are automatically called at the time of object creation.

❖ **Destructors** on the other hand ensure the de allocation of resources before an object dies or goes out of scope.

5

Lifecycle of an Object

❖ **Born Healthy**:

   ❖ Using **constructors**

❖ **Lives Safely:**

   ❖ Using **setters** and **getters**

❖ **Dies Cleanly:**

   ❖ Using **destructors**